要求辞書としての形式仕様詳細化手法の提案

幡 亮介[1]　大森 洋一[1]　荒木 啓二郎[1]　日下部 茂[2]

概要：近年，開発の上流工程で要求仕様を厳密に記述することで，ソフトウェアへの欠陥の混入を低減する形式手法と呼ばれる技術が注目されている．しかし，形式手法の利用には，自然言語仕様から形式仕様に落とし込む技術や形式化するスコープの規定など，開発者の習熟度に依存する問題がある．本研究では，STAMP/STPA を活用することで円滑に形式仕様の雛形を作成し，その後，開発を通して段階的に形式仕様の抽象度を下げていく詳細化手法を考案した．

1. はじめに

近年，IoT（モノのインターネット，Internet of Things）と呼ばれる概念に基づいた組込みソフトウェア開発が盛んにおこなわれている．あらゆるモノ同士が繋がることを想定したシステム開発が進むとともに，ソフトウェアは益々大規模・複雑化している．このような背景において，開発プロセスの見直しや新しい開発手法の活用が求められている．

しかし，開発を通して参照される仕様書や設計書といったドキュメントの書式は自然言語を用いることが多く，かつ多種多様である．このようなドキュメントの曖昧さは開発者同士のソフトウェアに対する認識に齟齬を生み，ソフトウェアに欠陥が混入する原因となる．

これに対し，開発の上流工程で要求仕様を厳密に記述することで内在する矛盾を排除する，形式仕様記述と呼ばれる技術がある．形式仕様記述を用いることで，開発早期での漏れの少ない要求仕様の作成が期待される．形式的な要求仕様の作成は，要求仕様の漏れや曖昧さに起因する不具合を低減することができるとされている [1]．

しかし，形式手法の利用には形式的に記述するスコープの定義や，自然言語仕様から形式仕様に落とし込む技術が必要となる．このような視点や技術には専門性やノウハウを求められることが多く，開発者には抽象化する能力が求められる [2]．一方で，スコープ定義には，STAMP/STPA を活用することが有効であるという主張もある [3]．

これらの点に注目し，本研究では STAMP/STPA を活用した形式仕様の雛形作成と，開発を通して形式仕様の抽象度を下げていく段階的な詳細化手法を提案する．

1 九州大学
Kyushu University, Fukuoka, Fukuoka 819–0395, Japan
2 長崎県立大学
University of Nagasaki, Sasebo, Nagasaki 858–8580, Japan

2. 詳細化手法

2.1 詳細化手法の概要

本研究で提案する詳細化手法では，文献 [4] に掲載されている V 字型モデルの 4 つの工程を通して、自然言語仕様から形式仕様に段階的に詳細化する（図 1）．本研究では，形式手法として VDM を用い，形式仕様記述言語として VDM-SL を採用する．

2.2 STEP1: 形式仕様の雛形作成

本研究では安全解析手法として STAMP/STPA を用いる．

STAMP モデリング工程では自然言語仕様から安全制約を識別し，コンポーネント間の制御関係を示すコントロール構造図を作成する（図 2）．コントロール構造図には，コンポーネントと制御アクションが記載される．形式仕様を記述する際には，コンポーネント名を module に，制御アクションの述語を operation あるいは function，制御アクションに含まれるその他の情報を type や state として記述する．矢印の向きは，機能や型の import および export と

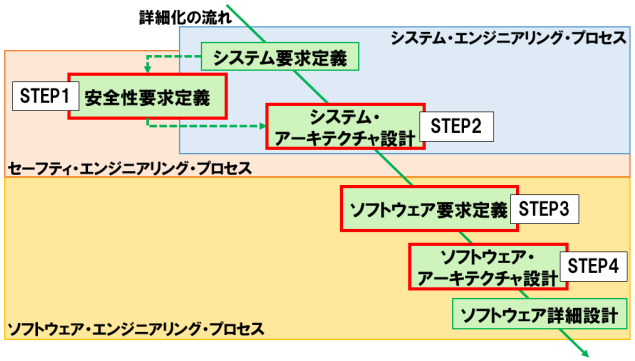

図 1　詳細化の手順

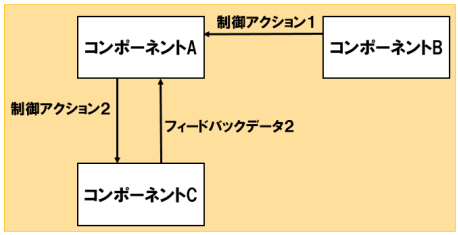

図 2　コントロール構造図の例

対応させる．

その後，STPA による安全解析をおこない，制約条件を追記する．安全制約が要求仕様以下の抽象度で守られていれば，STEP1 を終了する．守られていなければ，安全性を確保するために必要な機構や機能の追加または修正といった設計変更をおこなう [5]．設計変更後のシステムの合理性や各機能への影響は形式仕様を検査することで確認する．その後，安全解析を繰り返すことで，開発早期に要求仕様の漏れや制約条件を洗い出す．

2.3　STEP2: データ型の定義

機能ブロック図やデータ一覧表を作成する際に必要な機能やデータの識別をおこなう．その際に STEP1 で作成された形式仕様を要求辞書として活用する．要求辞書とは，用語辞書に加え，述語の定義も記された辞書である．データ一覧表を作成した後，形式仕様のデータ型を実装可能な抽象度まで詳細化する．

2.4　STEP3: 型定義と機能シグネチャの確認

STEP3 では形式仕様をソフトウェア機能要求リストとして活用し，型と機能の対応を確認する．また，フェイルセーフな機構や故障時に必要な機能が適切に設置されているか確認するために，形式仕様に記述されている機能や制約条件を参照する．追加する機能がある場合は，形式仕様に追記し検証することで内部整合性を保証する．

2.5　STEP4: 機能の詳細設計

ソフトウェア・アーキテクチャ設計工程では機能の詳細設計をおこなう．詳細化する機能を抽出する際に，形式仕様の機能定義を参照し，未定義な項目である is not yet specified や undefined 項目を詳細化する．

3.　適用事例

本稿では，STEP1 で抽象度の高い形式仕様を作成する際に，要求仕様を洗練できることの有効性を示す．具体的な要求仕様の例として，『LED キューブ装置 ソフトウェア要求仕様書』第 1 版を用いる [6]．

まず始めに，自然言語仕様から安全制約の識別とコントロール構造図（図 3）の構築を実施した．その後，コントロール構造図から形式仕様を作成した．

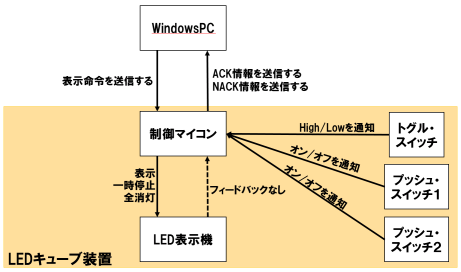

図 3　コントロール構造図　実施例

ガイドワード適用例

制御アクション	Not Providing	Providing causes hazard	Too early/ Too late	Stop too soon/ Applying too long
High/Low通知	モードを移行しない	–	High通知が遅延した場合，通知前の送信データを正常に受信できない	High通知が断続した場合，点灯動作がリセットされる
	・ ・ ・	・ ・ ・	・ ・ ・	・ ・ ・

図 4　ガイドワードと非安全な制御アクション抽出例

STPA を実施したところ（図 4），「High 通知が遅延した場合，通知前の送信データを正常に受信できない」という安全ではない状態に陥るアクションが明らかになった．そこで，一旦送信されたデータを保持する機構を作成するという対策を講じ，形式仕様に反映した．また，「High 通知が断続した場合，点灯動作がリセットされる」という非安全な制御アクションも発見した．これに対して，High 通知を受け取った場合，Low 通知を受信しない限り High の状態を保持する条件を付与した．

4.　おわりに

本稿では，提案する詳細化手法のうち，STEP1 を事例に適用した．その結果，形式仕様の作成に専門性やノウハウを必要とせずに非安全なアクションを発見でき，動的な振る舞いについても仕様の漏れや曖昧さを早期に発見できた．上記成果を踏まえ，形式手法と STAMP/STPA の併用はソフトウェアの高品質化に有望であると期待される．

参考文献

[1] Mitsubishi Research Institute, Inc.: ディペンダブルシステムのための形式手法の実践ポータル (online)，入手先 ⟨http://formal.mri.co.jp/⟩ (2010.12.01).
[2] 中島震：ソフトウェア工学の道具としての形式手法，国立情報学研究所 (2007).
[3] 日下部茂，荒木啓二郎：Pre-Formal メソッドとしての STAMP モデリング，第 1 3 回クリティカルソフトウェアワークショップ (2016).
[4] 情報処理推進機構 ソフトウェア・エンジニアリング・センター：改訂版 組込みソフトウェア向け開発プロセスガイド，翔泳社 (2007).
[5] 八山幸司：米国における STAMP（システム理論に基づく事故モデル）研究の最新の動向，ニューヨークだより (2015).
[6] Interface：LED キューブ装置 ソフトウェア要求仕様書 第 1 版 (online)，入手先 ⟨http://www.cqpub.co.jp/interface/download/2011/03/SpecOfLedcube_Ver1.PDF⟩ (2011.01.25).